Electronic Version
Stylesheet Version v1.1.1

Description

METHOD OF SHARING STATE BETWEEN STATEFUL INSPECTION FIREWALLS ON MEP NETWORK

BACKGROUND OF INVENTION

[0001] Field of the Invention

[0002] The present invention relates generally to a method of sharing a state between stateful inspection firewalls on a multiple entry/exit point network and, more particularly, to a method of sharing a state between stateful inspection firewalls on a multiple entry/exit point network, which enables the state to be shared between the stateful inspection firewalls using a modified SYN cookie on the multiple entry/exit point network having a plurality of access points physically remote from each other.

[0003] DESCRIPTION OF THE RELATED ART

[0004] In general, a firewall is located at the boundary of a network, and functions to protect the network from the out-

side thereof. Recently, of various firewalls, a stateful inspection firewall is widely used. The stateful inspection firewall performs the function of a firewall in such a way as to intercept an incoming or outgoing packet, extract connection information, such as the source address, destination address, protocol, source port number and destination port number of the packet, from the packet, update a state table, and makes the determination of filtering based on the updated state table.

[0005] With reference to the accompanying drawings, the operation of a conventional stateful firewall 30 is described in detail below.

[0006] Fig. 1 is a system configuration diagram showing the operation of the conventional stateful inspection firewall 30.

[0007] As shown in Fig. 1, the stateful inspection firewall 30 is located between a client 10 and a server 20, and data are exchanged between the server 20 and the client 10 according to the Transmission Control Protocol (TCP). That is, data are exchanged between the server 20 and the client 10 according to the '3-way handshaking' rule.

[0008] In accordance with the '3-way handshaking' rule, there are performed the first step of the client 10 sending a SYN packet requesting an access to the server 20, the second

step of the server 20 sending a SYN/ACK packet indicating the acceptance of the request to the client 10, and the third step of the client sending an ACK packet to the server 20, a connection being established between the server 20 and the client 10 and data being exchanged between the server 20 and the client 10.

[0009] Fig. 2 is a diagram showing the format of a TCP header.

[0010] A SYN packet, a SYN/ACK packet and an ACK packet are determined by the TCP header. With reference to Fig. 2, the SYN packet is determined when a SYN flag 50 is 1 and an ACK flag 52 is 0, the SYN/ACK packet is determined when the SYN flag 50 is 1 and the ACK flag 52 is 1, and the ACK packet is determined when the SYN flag 50 is 0 and the ACK flag 52 is 1. Furthermore, each of the packets includes a sequence number 54 and an acknowledgement number 56, in which the sequence number 54 of the SYN packet and the SYN/ACK packet becomes an Initial Sequence Number (ISN). The sequence number of the SYN packet, which the client 10 sends to the server 20 at the first step of the '3-way handshaking' rule, becomes ISN_c, and the sequence number 54 of the SYN/ACK packet, which the server 20 sends to the client 10 at the second step thereof, becomes ISN_s. In the meantime, the ac-

knowledgement number 56 becomes ISN_c+1 in the SYN/ACK packet that the server 20 sends to the client 10, and becomes ISN_s+1 in the first ACK packet that the client 10 sends to the server 20.

[0011] In Fig. 1, when the client 10 sends the SYN packet to the server 20 while requesting an access to the server, the firewall 30 inspects the SYN packet, and passes the SYN packet therethrough if such a connection is set to be permitted. The firewall 30 should pass therethrough the SYN/ACK packet, which is sent from the server 20 to the client 10 in response to the SYN packet, as well as the SYN packet, which the client 10 sends while requesting the access to the server 20. This can be implemented by recording connection information in the state table of the firewall 30. The firewall 30 searches the connection information of the state table, and passes the packet therethrough if corresponding connection information exists.

[0012] Fig. 3 is a diagram showing the state table of the conventional firewall 30. In the state table t can be recorded connection information, including a source address t1, a destination address t2, a protocol t3, a source port number t4, a destination port number t5 and a connection state

t6.

[0013] When the client 10 sends the SYN packet to the server 20 while requesting an access to the server 20, the firewall 30 extracts the source address t1, the destination address t2, the protocol t3, the source port number t4, and the destination port number t5 from the SYN packet, records the extracted information in the state table t, and records the connection state t6 as 'SYN_SENT.' Thereafter, when the SYN/ACK packet in response to the SYN packet arrives, the firewall 30 searches the state table t for connection information related to such a connection, and passes the SYN/ACK packet therethrough if the connection information exists. Subsequently, the firewall 30 changes the connection state t6 to 'SYN_RECV' because the firewall 30 has received the SYN/ACK packet, and then passes the SYN/ACK packet therethrough. In brief, the stateful inspection firewall 30 performs the function of a firewall by keeping track of the connection state t6 and recording it.

[0014] However, the conventional stateful inspection firewall is problematic in that it is only available on a network having a single entry point because all the incoming and outgoing traffics of a connection must be monitored to keep track of the connection state t6. That is, the conventional

stateful inspection firewall 30 is operable only on a Single Entry Point (SEP) network, but is not operable on a MEP network having a plurality of entry points because an outgoing traffic and an incoming traffic may be passed through different firewalls, and thus it is difficult to keep track of the state.

SUMMARY OF INVENTION

[0015] Accordingly, the present invention has been made keeping in mind the above problems occurring in the prior art, and an object of the present invention is to provide a method of sharing a state between stateful inspection firewalls on an MEP network, which enables the state to be shared between the stateful inspection firewalls physically remote from each other using a modified SYN cookie (hereinafter referred to as a "m.SYN cookie") when data is exchanged according to the '3-way handshaking' rule.

[0016] In order to accomplish the above object, the present invention provides a method of sharing a state between stateful firewalls on an MEP network for data exchange between a server and a client through firewalls physically remote from each other, comprising the steps of (a) one of the firewalls receiving a SYN packet sent from the client to the server; (b)the firewall creating an m.SYN cookie,

modifying the SYN packet using the m.SYN cookie and sending the SYN packet to the server, and the server sending a SYN/ACK packet to the client in response to the SYN packet; (c) the firewall, which has received the SYN/ACK packet, extracting a firewall identifier ID_{fw} from the SYN/ACK packet and sending the SYN/ACK packet to a corresponding one of the firewalls, the corresponding firewall searching a state table for connection information and sending the connection information, together with the SYN/ACK packet, to the firewall, which has received the SYN/ACK packet; and (d) the firewall, which has re-received the SYN/ACK packet, updating the state table, changing an acknowledgement number of the SYN/ACK packet to an Initial Sequence Number (ISN_c)+ 1, and sending the SYN/ACK packet to the client.

Brief Description of Drawings

[0017] The above and other objects, features and advantages of the present invention will be more clearly understood from the following detailed description taken in conjunction with the accompanying drawings, in which:

[0018] Fig. 1 is a system configuration diagram showing the operation of a conventional stateful inspection firewall.

[0019] Fig. 2 is a diagram showing the format of a TCP header.

[0020] Fig. 3 is a diagram showing the state table of the conventional firewall.

[0021] Fig. 4 is a system configuration diagram illustrating a method of sharing a state between stateful inspection firewalls on an MEP network in accordance with the present invention.

[0022] Fig. 5 is a block diagram of a stateful inspection firewall in accordance with the present invention.

[0023] Fig. 6 is a flowchart showing the method of sharing the state between the stateful inspection firewalls on the MEP network.

[0024] Fig. 7 is a diagram showing an m.SYN cookie in accordance with the present invention.

[0025] Fig. 8 is a diagram showing the state table t of the stateful inspection firewall in accordance with the present invention.

DETAILED DESCRIPTION

[0026] Reference now should be made to the drawings, in which the same reference numerals are used throughout the different drawings to designate the same or similar components.

[0027] Fig. 4 is a system configuration diagram illustrating a method of sharing a state between stateful inspection

firewalls on an MEP network in accordance with the present invention.

[0028] The MEP network, as shown in FIG. 4, includes a client 10, a server 20, and a firewall 1 30a and a firewall 2 30b that are physically remote from each other. In this case, the firewall 1 30a and the firewall 2 30b are installed to protect the network of the client 10 from the outside thereof. The firewall 1 30a and the firewall 2 30b are stateful inspection firewalls 30, which intercept exchanged packets, extract connection information from the intercepted packets, update internal state tables t, and make the determination of filtering based on the updated state tables t.

[0029] Fig. 4 depicts only a preferred embodiment of the present invention for an illustrative purpose. Although the method of sharing the state between the stateful inspection firewalls on the MEP network according to the present invention can be applied to the case where a client is located outside and a server is located inside, etc., the same inventive concept is employed, so that only the case of FIG. 4 is described in detail below.

[0030] In Fig. 4, in order to enable data to be exchanged between the client 10 and the server 20, a traffic outgoing from the

network of the client 10 to the server 20 and a traffic incoming from the server 20 to the network of the client 10 should pass through the firewall 30. At this time, the case where the outgoing and incoming traffics pass through the same firewall does not matter. The case where the outgoing and incoming traffics pass through different firewalls (asymmetrical paths) requires the sharing of a state between the firewall 1 30a and the firewall 2 30b.

[0031] Fig. 5 is a block diagram of a stateful inspection firewall 30 in accordance with the present invention.

[0032] As shown in Fig. 5, the firewall 30 includes a communications module 310, a control module 320 and a database 330.

[0033] The communications module 310 functions to receive and send packets. The control module 320, as shown in Figs. 5 and 6, functions to control the execution of processes related to the method of sharing a state between stateful inspection firewalls on an MEP network.

[0034] In more detail, the control module 320 includes a packet verifying module 321 verifying whether a received packet is valid or invalid according to a firewall rule set by an administrator, an m.SYN cookie creating module 322 creating an m.SYN cookie, a packet modifying module 323

modifying the packet according to a set process, a state table updating module 324 updating a state table t according to the set process, a search module 325 searching the state table t for connection information and searching information stored in the database 330, and an m.SYN cookie verifying module 326 verifying whether m.SYN cookie is valid.

[0035] The database 330 includes a firewall identifier (hereinafter referred to as a "ID_{fw}") i, a state table t storing connection information, a time counter c, and a secret key k. The ID_{fw} i is a bit value identifying each of the firewalls included in the network, the state table t is the table in which the connection information of the firewall 30 is stored, and the time counter c is a bit counter that is included in the firewall 30 and increased at certain intervals. Furthermore, in the database 330 is included the secret key k unique to the network.

[0036] The method of sharing the state between stateful inspection firewalls 30 on the MEP network uses an m.SYN cookie to allow the state to be shared between the firewall 1 30a and the firewall 2 30b that are physically remote from each other when data are exchanged according to the '3-way handshaking' rule. While it is assumed that the

firewall creating the m.SYN cookie is set to the firewall 1 30a, the firewall verifying the m.SYN cookie is set to the firewall 2 30b and all the firewalls 30 share the synchronized time counter c increasing every 16 seconds, the method of sharing the state between the stateful inspection firewalls is described in detail below.

[0037] Fig. 6 is a flowchart showing the method of sharing the state between the stateful inspection firewalls 30 on the MEP network.

[0038] With reference to Fig. 6, the client 10 sends a SYN packet to the firewall 1 30a at step S10. The firewall 1 30a receives the SYN packet through the communications module 310, and the packet verifying module 321 verifies whether the SYN packet is valid according to a firewall rule set by an administrator at step S20. If, as a result of the verification, the SYN packet is not valid ('N' at step S20), and the SYN packet is discarded in the firewall 1 30a at step S25. If the SYN packet is valid ('Y' at step S20), the m.SYN cookie creating module 322 creates the m.SYN cookie at step S28.

[0039] Fig. 7 is a diagram showing the m.SYN cookie 40 that is created in the m.SYN cookie creating module 322.

[0040] As shown in Fig. 7, the m.SYN cookie 40 includes ISN_{17}

42, T_0 44and '$Hash_{13}$+ID_{fw}' 46.

[0041] The ISN_{17} 42 is determined by the upper 17 bit value of ISN of the SYN packet to support fast reincarnation.

[0042] In regard to the reincarnation of a TCP connection, there is the prescription "assigns its ISN for the new connection to be larger than the largest sequence number it used on the previous connection incarnation."

[0043] In the present invention, the fast reincarnation of a TCP connection does not occur frequently. If the fast reincarnation occurs, it is assumed that ISN increases to be larger than SNprev (the largest sequence number it used on the pervious connection incarnation) by at least 32768.

[0044] In more detail, the fact that ISN is larger than SNprev by at least 32768 (2^15) imports that the 16-th bit of a 32-bit binary number is larger by 1 in terms of a bit level. Consequently, in the host supporting fast reincarnation, the upper 17 bit value (ISN_{17} 42) of the ISN of the SYN packet is larger than the upper 17 bit value of the SNprev by at least 1 on a bit level.

[0045] If the ISN fulfills the above-described preconditions, m.SYN cookie 40 is larger than SNprev even though any numerical value is inserted into the lower 15 bits in addition to ISN_{17} 42. Accordingly, in the SYN packet in which

the ISN has been replaced with the m.SYN cookie 40, the ISN is larger than the SNprev, so that the method of sharing the state between the stateful inspection firewalls 30 on the MEP network can support a host in which fast reincarnation occurs.

[0046] Furthermore, in the method of sharing the state between the stateful inspection firewalls 30 in accordance with the present invention, the firewalls 30, which are the subjects of the creation and verification of the m.SYN cookie 40, may be different from each other, so that T_0 44 is included in the m.SYN cookie 40. The T_0 44 is the least significant two bits of time $time_{org}$ indicated by the time counter c when the firewall 1 30a creates the m.SYN cookie 40, and is defined by the following Equation 1. With the Equation 1, the firewall 2 30b accurately extracts the time when the m.SYN cookie 40 is created, and can use the extracted value as an input to a hash function inspecting whether the m.SYN cookie 40 is valid.

[0047] $$T_0 = time_{org} \bmod 4 \quad (1)$$

[0048] where $time_{org}$ is the time indicated by the time counter c when the firewall 1 30a creates the m.SYN cookie 40, and mod4 is the remainder obtained through division by 4.

[0049] Furthermore, the m.SYN cookie 40 includes '$Hash_{13}+ID_{fw}$'

46. In the present invention, $Hash_{13}$ is determined by the following Equation 2, and is 13 bits, unlike the fact that the output value of the hash function of a conventional SYN cookie is 32 bits.

[0050] $Hash_{13} = Hash(k, sa, sp, da, dp, time_{org}, ISN_c>>15) \% 2^{13}$ (2)

[0051] where Hash() is the output value of a hash function, k is a secret key, sa is a source address t1, sp is a source port number t4, da is a destination address t2, dp is a destination port number t5, $ISN_c>>15$ is a value obtained by eliminating the lower 15 bits from ISN_c, and Hash()%2^13 is the value of the lower 13 bits of the output value of the hash function.

[0052] As shown in the Equation 2, in the present invention, $Hash_{13}$ is determined using the secret key k shared by the firewalls 30 as a variable of the hash function. Accordingly, only if the firewall 2 30b learns the secret key k, the firewall 2 30b can produce the same $Hash_{13}$ at the time of verification. That is, the secret key k is used to prevent an attacker from counterfeiting the m.SYN cookie. Since attackers do not know the secret key k, most of the counterfeited m.SYN cookies are discarded during verification even though the attackers randomly produce the m.SYN

cookies. Meanwhile, '$Hash_{13}+ID_{fw}$' 46, which is the last 13 bits of the m.SYN cookie 40, is finally determined by adding the firewall identifier to the $Hash_{13}$.

[0053] Referring to Fig. 6 again, the m.SYN cookie creating module 322 of the firewall 1 30a creates the m.SYN cookie 40 including the above-described values at step S28. Thereafter, the packet modifying module 323 of the firewall 1 30a replaces the ISN_c of the received SYN packet with the m.SYN cookie 40, and the state table updating module 324 updates the connection information of the state table t (source address, source port number, destination address, destination port number, and the difference between the ISN_c and the m.SYN cookie) at step S30. In this case, the updated state table t is stored in the database 330.

[0054] Fig. 8 is a diagram showing the state table t of the stateful inspection firewall 30 in accordance with the present invention.

[0055] Referring to Fig. 8, the state table t includes 'm.SYN cookie-ISN_c' t7, in addition to the items of the conventional state table t. The 'm.SYN cookie-ISN_c' t7 functions to allow the firewall 2 30b to learn the value of the ISN_c even though the firewall 1 30a replaces the ISNc of the

SYN packet with the m.SYN cookie 40.

[0056] After the packet modifying module 323 of the firewall 1 30a replaces the ISN_c of the SYN packet with the m.SYN cookie 40 and the state table updating module 324 updates the connection information of the state table t of the firewall 1 30a at step S30, the modified SYN packet is sent to the server 20 through the communications module 310 at step S40. Subsequently, the server 20 sends a SYN/ACK packet to the client 10 in response to the SYN packet at step S50. At this time, the acknowledgement number 56 of the SYN/ACK packet becomes 'm.SYN cookie + 1.'

[0057] In the meantime, the SYN/ACK packet sent from the server 20 to the client 10 reaches the firewall 2 30b prior to reaching the client 10. When the communications module 310 of the firewall 2 30b receives the SYN/ACK packet, the m.SYN cookie verifying module 326 of the firewall 2 30b is activated. The m.SYN cookie verifying module 326 acquires the ID_{fw} from the m.SYN cookie 40, which is extracted from the acknowledgement number 56 of the SYN/ACK packet, through the use of the following Equation 3 at step S62.

[0058] $ID_{fw} = (SC\text{-}Hash(k, sa, sp, da, dp, time_{input}, SC>>15)) \%$

2^13 (3)

[0059] where SC is the m.SYN cookie 40 extracted from the acknowledgement number 56 of the SYN/ACK packet, SC>>15 is the value obtained by eliminating lower 15 bits from the SC, and ()% 2^13 is the lower 13 bits of value of ().

[0060] In the Equation 3, $time_{input}$ is obtained from the following Equation 4.

[0061] $time_{input} = time_{curr} + 1 - ((time_{curr} + 1\ (SC >> 13))\ mod4)$

[0062] $= time_{curr} + 1-((time_{curr}+1-T_0)\ mod4)$ (4)

[0063] where $time_{curr}$ is the time indicated by the time counter c of the firewall 2 30b at the time of verifying the m.SYN cookie, and SC >> 13 is the value obtained by eliminating lower 13 bits from the SC.

[0064] The m.SYN cookie verifying module 326 extracts ID_{fw} using the Equations 3 and 4 at step S62, and verifies whether the extracted ID_{fw} is valid at step S63. In this case, if the extracted ID_{fw} does not fulfill "$0 \leq ID_{fw} \leq MAX_{id}$ (MAX_{id}: the greatest value of the ID_{fw}s of the firewalls)" ('N' at step 63), the m.SYN cookie 40 was counterfeited and the received packet is discarded. If the extracted ID_{fw} fulfills "$0 \leq ID_{fw} \leq MAX_{id}$" ('Y' at step 63), the process pro-

ceeds to the next step.

[0065] If the extracted ID_{fw} is verified to be valid ('Y' at step S63), the m.SYN cookie verifying module 38 compares the extracted ID_{fw} with its own ID_{fw} at step S64. If, as a result of the comparison, the extracted ID_{fw} is identical with the ID_{fw} of the m.SYN cookie verifying module 38 ('Y' at step S64), the state table updating module 324 searches the state table t for connection information. If the connection information exists ('Y' at step S65), the state table updating module 324 updates the state table t to allow 'SYN_RECV' to be recorded in the connection state t6. The packet modifying module 36 changes the acknowledgement number 56 of the SYN/ACK packet to 'ISN_c + 1.' In this case, the ISN_c is the value obtained by subtracting the 'm.SYN cookie-ISN_c' t7 from the m.SYN cookie 40, so that the firewall 2 30b can learn the ISN_c at step S70.

[0066] In the meantime, if the extracted ID_{fw} is different from the ID_{fw} of the firewall 2 30b (that is, asymmetrical paths), the communications module 310 sends the SYN/ACK packet to the firewall 1 30a corresponding to the extracted ID_{fw} at step S66.

[0067] The search module 325 of the firewall 1 30a having received the SYN/ACK packet searches the state table t for

the connection information at step S67. If the connection information exists ('Y' at step S67), the search module 325 updates the connection state t6 of the state table t of the firewall 1 30a as 'SYN_RECV' and sends the connection information, together with the SYN/ACK packet, to the firewall 2 30b at step S68.

[0068] Thereafter, the state table updating module 324 of the firewall 2 30b updates the state table t so that 'SYN_RECV' is recorded in the connection state t6 of the state table t, and the packet modifying module 323 replaces the acknowledgement number 56 of the SYN/ACK packet with 'ISN_c + 1' at step S70.

[0069] Thereafter, the modified SYN/ACK packet is sent to the client 10 through the communications module 310 of the firewall 2 30b at step S80, so that the connection information can be shared between the firewall 1 30a and the firewall 2 30b. With this, the following packets, including the next ACK packet, can be directly passed through the two firewalls without additional information exchange.

[0070] In the meanwhile, the method of sharing the state between the stateful inspection firewalls according to the present invention can be applied to the case where a firewall and a Network Address Translator are used together,

and a File Transfer Protocol connection, besides the above-described embodiment.

[0071] Although the preferred embodiments of the present invention have been disclosed for illustrative purposes, those skilled in the art will appreciate that various modifications, additions and substitutions are possible, without departing from the scope and spirit of the invention as disclosed in the accompanying claims.